SPECIFICATION AMENDMENTS

Please replace the paragraph beginning on page 4, line 15 with the following replacement paragraph:

As is known in the art, to program the configuration memory 20, a user couples an external programming tool to configuration engine 10. During the subsequent configuration, configuration engine 10 sequentially addresses all memory locations within configuration memory 20 so that the appropriate configuration data may be written into these locations. The addresses and configuration data couple from configuration engine ~~20~~ 10 to configuration memory ~~10~~ 20 over a system bus 40.

Please replace the paragraph beginning on page 4, line 22 with the following replacement paragraph:

Once all the configuration data has been written into the configuration memory, the programmable logic device 5 may commence ~~operation,~~ operation by implementing the logical functions desired by a user. Should the configuration data become corrupted, the subsequent operation of the programmable logic device may fail to satisfy a user's requirements. Thus, during operation of the programmable logic device, self-verify control module 30 controls the continuous verification of the configuration data stored in configuration memory ~~10~~ 20 to detect any corruption. The stored configuration data are verified by processing with an error detection algorithm. Should an error be detected, self-verify control module 30 signals the faulty condition.

Please replace the paragraph beginning on page 6, line 4 with the following replacement paragraph:

To verify its integrity, configuration data written into configuration memory 20 is retrieved during operation of the programmable logic device. Because configuration engine 10 uses system bus 40 to load the configuration data into configuration memory 20 during configuration, it is efficient to also use system bus 40 to retrieve the configuration data during self-verification. It will be appreciated, however, ~~that,~~

~~alternatively,~~ that a separate bus may be used to retrieve the configuration data so that the error detection analysis may be performed.

Please replace the paragraph beginning on page 9, line 21 with the following replacement paragraph:

To begin a self-verification cycle, self-verify control module 30 (Figure 1) signals a counter 205 included in module 50 to begin counting through all available addresses for configuration memory 20. Responsive to each address from counter 205, the corresponding configuration data word is retrieved from configuration memory 20 on bus 40. As described above, configuration memory cells that may be reconfigured during operation of the programmable logic device should be excluded from the CRC calculation. Denoting which configuration memory cells should be excluded may occur in a number of ways. Should the denotation be made through use of a status bit, a test module 220 may test each status bit to indicate whether LFSR 200 should process the associated configuration data word. If the status bit indicates the associated configuration data word should not be processed, test module 220 blocks the word from coupling to LFSR 200. In this case, LFSR 200 simply maintains its current value. Should the status bit indicate that the associated configuration data word should be processed, test module 220 allows it to couple to LFSR 200. Note, however, that "blocked" configuration data may couple assuming that predetermined values are used. For example, the blocked configuration data may be considered to be all logical high values or logical low values. By successively processing each retrieved data word that couples through test module 220 as counter 205 cycles through every address in configuration memory 20, LFSR 200 eventually stores the CRC checksum for the configuration data. Counter 205 then indicates to self-verify control module 30 that all addresses have been counted so that LFSR 200 may be commanded to provide the CRC checksum to CRC comparator 60. CRC comparator 60 then compares the CRC checksum to the golden CRC checksum retrieved from storage register 240. If the golden CRC checksum and the CRC checksum are the same, a device_good flag is set to true and stored in a flag register 251 within self-verify control module 30. Otherwise, the device_good flag is set to false and then stored in flag register 251 ~~250~~. The status of device_good flag may then be reported by module

30 to external devices so that a reconfiguration of the programmable device may be initiated. For example, should the external devices include a microprocessor, an interrupt could be generated. Should programmable logic device 5 also include a non-volatile memory storing the configuration data, it could react to a bad device_good flag by automatically reloading the volatile configuration memory 20 from the non-volatile memory.

Please replace the paragraph beginning on page 13, line 16 with the following replacement paragraph:

ROM error flag 360 and configuration error flag ~~310~~ 325 may be received by external devices so that programmable logic device 5 may be reconfigured. Alternatively, these flags may be OR'd together before reporting the results to an external device. In addition, these flags may be used internally by programmable logic device 5. For example, should programmable logic device 5 contain a non-volatile configuration memory, it may respond to a faulty condition expressed by either of these flags by reconfiguring volatile configuration memory 20 from the non-volatile configuration memory.